TITLE OF THE INVENTION

CHECKSUM WRITING METHOD AND CHECKSUM CHECKING APPARATUS

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application claims the benefit of Korean Patent Application No. 2003-8459, filed on February 11, 2003, in the Korean Intellectual Property Office, the disclosure of which is incorporated herein in its entirety by reference.

BACKGROUND OF THE INVENTION

1. Field of the Invention

[0002] The present invention relates to a checksum writing method and a checksum checking apparatus, and more specifically, to a method of calculating and writing a checksum for a memory and an apparatus which checks a checksum used in a system having a microcomputer and a memory.

2. Description of the Related Art

[0003] Generally, an apparatus that uses a microcomputer also uses a programmable storage medium, such as a ROM or flash memory. When a problem arises in an apparatus or it is necessary to check information on programs in the memory, a checksum is checked.

[0004] Previously, in order to check if the checksum was correct, the apparatus had to be dismantled and a checksum value printed on the outside of the memory had to be verified. A considerable amount of time is required to dismantle and reassemble an apparatus, and moreover the apparatus or product may be damaged.

[0005] Therefore, it would be beneficial to write and check a checksum in a memory without dismantling the associated apparatus.

SUMMARY OF THE INVENTION

[0006] The present invention provides a method of calculating a checksum of data stored in a memory and writing the calculated checksum in a predetermined area of the memory, and an apparatus for performing the method.

[0007] According to an aspect of the present invention, there is a method of calculating and writing a checksum in a memory, the method comprising: calculating a first checksum by reading values from the memory using a predetermined unit and summing the read values; calculating a first mode checksum by subtracting values written in a predetermined area of memory from the first checksum; initializing a second checksum to be zero if the first mode checksum does not meet a predetermined condition; calculating a second mode checksum by inverting the second checksum and adding the inverted second checksum to the first mode checksum; and writing the inverted second checksum value in the predetermined area of the memory, if the second mode checksum is equal to the second checksum.

[0008] The terms "invert" and "inverted" as used above, and throughout the specification and claims, refer to inverting the sign of a value by multiplying the value by -1. For example, using the terms as here defined, the inverted value of X is -X, and the inverted value of -X is X.

[0009] According to another aspect of the present invention, there is an apparatus for checking a checksum comprising: a memory that stores a predetermined program and a checksum in the memory; a microcomputer that performs the program stored in the memory, inverts the checksum stored in the memory and outputs the inverted checksum from the memory; and a checksum calculator that computes and writes a checksum in the memory by: calculating a first checksum by reading values from the memory using a predetermined unit and summing the read values; calculating a first mode checksum by subtracting values written in a predetermined area of the memory from the first checksum; initializing a second checksum to be zero if the first mode checksum does not meet a predetermined condition; calculating a second mode checksum by inverting the second checksum and adding the inverted second checksum to the first mode checksum; and writing the inverted second checksum value in the predetermined area of the memory, if the second mode checksum is equal to the second checksum.

[0010] Additional aspects and/or advantages of the invention will be set forth in part in the description which follows and, in part, will be obvious from the description, or may be learned by practice of the invention.

BRIEF DESCRIPTION OF THE DRAWINGS

[0011] These and/or other aspects and advantages of the invention will become apparent and more readily appreciated from the following description of the embodiments, taken in conjunction with the accompanying drawings of which:

[0012]

FIG. 1 is a block diagram showing a checksum checking apparatus according an embodiment of to the present invention;

FIG. 2 is a flow chart showing a general checksum calculation method;

FIG. 3 shows data stored at the end portion of a memory; and

FIG. 4 is a flow chart showing a checksum writing method according an embodiment of to the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0013] Reference will now be made in detail to the embodiments of the present invention, examples of which are illustrated in the accompanying drawings, wherein like reference numerals refer to the like elements throughout. The embodiments are described below to explain the present invention by referring to the figures.

[0014] Hereinafter, a checksum calculation method and a checksum checking apparatus according to embodiments of the present invention will be described in detail with reference to the attached drawings.

[0015] FIG. 1 is a block diagram of a checksum checking apparatus according to an embodiment of the present invention. The checksum checking apparatus shown in FIG. 1 includes a device 10 having a microcomputer 10-1, a memory 10-2 (shown here, by way of example, as ROM), a checksum calculator 10-3, and a display 11. The microcomputer 10-1 reads programs stored in the memory 10-2 and performs operations corresponding to those

programs. The checksum calculator 10-3 reads data stored in the memory 10-2, calculates a checksum, and stores the calculated checksum in a predetermined area of memory 10-2. The microcomputer 10-1 can read a checksum stored in the memory 10-2 and display the checksum on the display 11 outside the device 10 at the request of a user. Here, light emitting diodes (LEDs) or an on-screen-display (OSD) can be used as the display 11.

[0016] The operation of the checksum calculator 10-3 will now be explained by referring to the checksum calculation method shown in FIGS. 2 and 4. FIG. 3 shows the end portion of a memory with data stored therein.

[0017] The checksum calculator 10-3 calculates a checksum by reading data stored in the memory byte by byte. In operation 20 in FIG. 2, a ROM pointer (rom_ptr) indicating a memory address is set to zero and a checksum (chksum) is initialized to zero. For reference, numbers beginning with 0x or ending with "h" in FIGS. 2 through 4 indicate hexadecimal values. In operation 21, a new checksum (chksum) is calculated by adding an existing checksum value (chksum) to a value (*rom_ptr) written in an address pointed to by the ROM pointer. Next, in operation 22, a value of the ROM pointer is increased by one. Then, in operation 23, if the increased value of the ROM pointer is less than or equal to the total size of the memory, operations 21 through 23 are repeated.

[0018] If the increased value of the ROM pointer is greater than the total size of the memory, the checksum calculated in the previous operation is stored in a predetermined area of the memory 10-2, for example, in the last two bytes, A and B, as shown in FIG. 3. In FIG. 3, reference numerals 30 and 31 indicate an address and a data-written portion, respectively. Typically, the end portion of the memory shown in FIG. 3 is empty. Therefore, if a checksum is written at the last two bytes of memory, the checksum can be checked without dismantling the apparatus.

[0019] However, writing the checksum in the last two bytes of the memory causes the internal checksum value for the entire memory to be changed. Therefore, several additional bytes are needed to correct the changed checksum value.

[0020] Accordingly, a method of writing a checksum without changing the checksum for the entire memory is required. FIG. 4 is a flow chart showing such a method. A first checksum (chksum) of the whole memory is calculated in operation 40 according to the method of FIG. 2.

Next, a first mode checksum (mod1_chksum) is calculated by subtracting the checksum calculated in FIG. 2 (2*(0xFF)), stored as an example in A and B of FIG. 3, from the calculated first checksum in operation 41. Then, in operation 42, if modulo-calculating the first mode checksum (mod_chksum) by 257 (0x0101) results in a value of 2, the least significant bit of the byte expressed as C in FIG. 3 is corrected in operation 43. For example, if a value stored in C is 0xFF, it is corrected to be 0xFE. This indicates that every 258th data value, starting from 0, out of a total of 65536 values should be corrected. In operation 44, a second checksum (i_chksum), initially having a value of 0, is increased by one. The second checksum (i_chksum) is then inverted. Then, as shown in operation 45 in FIG. 4, a higher byte value of the inverted second checksum is substituted for the value in A and a lower byte value of the same checksum is substituted with the value stored in B. In operation 46, the byte values stored in A and B are added to the first mode checksum of operation 41 and this added value is defined as a second mode checksum (mod2_chksum) in order to be distinguished from the first mode checksum of operation 41. In operation 47, if the second checksum (i_chksum) is the same as the second mode checksum (mod2_chksum) the inverted second checksum is written to A and B in operation 48. If the above two values are not the same in operation 47, the present second checksum value is increased by one and operations 45 to 47 are repeated until the second checksum becomes 0xFFFF.

[0021] If a checksum is checked later, the inverted second checksum written in A and B of the memory is read and then inverted to check if the checksum is normal. That is, the result obtained by check-summing the inverted second checksum (-i_chksum) written in the last two bytes and values written in bytes other than the last two bytes of memory becomes the second checksum (i_chksum). Thus, later checksum checking can be done by re-inverting the inverted second checksum written in the last two bytes and outputting the re-inverted value. This checksum value can be shown on the display 11 of FIG. 1.

[0022] According to the present invention, a checksum can be written and then checked more easily without dismantling the apparatus, which allows for more convenient writing and checking of the checksum, since two bytes, or only two bytes plus one bit for 256 cases among empty portions in the memory are additionally required when writing the checksum in the memory.

[0023] The present invention may be embodied as a computer code, which can be read by a computer, on a computer readable recording medium. The computer readable recording medium includes all manner and types of recording apparatuses on which computer readable data are stored.

[0024] The computer readable recording media includes at least storage media such as magnetic storage media (e.g., ROM's, floppy disks, hard disks, etc.), optically readable media (e.g., CD-ROMs, DVDs, etc.), and carrier waves (e.g., transmissions over the Internet). Also, the computer readable recording media can be distributed to computer systems connected through a network and can be stored and executed as a computer readable code in a distributed mode.

[0025] Although a few embodiments of the present invention have been shown and described, it would be appreciated by those skilled in the art that changes may be made in this embodiment without departing from the principles and spirit of the invention, the scope of which is defined in the claims and their equivalents.